THOUGHT UP IN CANADA AKA HOBBY

TS Bulletin #2-92

By, WILLIAM HARMER
97 RUSKIN STREET
OTTAWA, CANADA
K1Y 4B3

FOR FURTHER INFORMATION CONTACT A USER GROUP

Issue#2-92 --- © June, 1992 in the public domain, copy freely ---June 30, 92--8 pages--

This issue of the TS Bulletin contains an answer to a question that even we amateur programmers sometimes wonder about. How does a big database like dBASE III (Borland) work inside? Part of the answer you can find on our page 2 where a list of a routine in ZX-81/TS1000 BASIC shows an indexed sort, one in which pointers, N(X) to data items (in array I$(X)) are sorted to put them into alphabetical order, rather than moving the data items themselves around. That can be done on a further command like the dBASE III, PACK command. Sorting an array of numerical pointers (index sort) is to be used instead, where time is of the essence. Of course in the type of database that you will be using on a Timex-Sinclair computer, you are unlikely to get it so big that it can't be sorted quickly by ordinary means, but this sort is just something to add to your collection of sorting routines for educational reasons.

The question, "What is a computer, really?", is tackled on pages 6 and 7, since there seems to be a new myth in the making now circulating, to quash. Page 4 is another in our series of famous computers and one that may contain the answer of where the idea of the two register structure of the 6800, an early microprocessor, came from. Was it the Hewlett-Packard 2000 series of minicomputers or did they both have a common inspiration somewhere else? The hardware hacker is not neglected and on page 5 some printed circuit board making techniques are discussed, a subject seen in hobby publications recently. Talk about alien, some of the methods behind reading MS DOS 5¼ inch DSDD 360K disks and others are discussed by way of introducing some of the specs of this famous disk standard. And don't forget to look at page 8. All, in all, another issue of tidbits culled from many sources, to please the TS hobbyist, if said hobbyist has a wide-ranging mind. This issue is somewhat parallel to our QL Bulletin #1-1992, for which it is hoped the reader will forgive any duplications, now or in the future, but some material is too interesting not to share.

CONTENTS:
-Indexed Sorting Routine (ZX-81/TS1000 BASIC). p.2
-From Our ZX-81 LDOS Manual-Disk Words Quickly Explained p.8
-What Is A Computer? An Old Question Resurfaces (2 parts) ... p.7, p.6
-Hardware Tricks— A Few Amateur Hackers' PC Board Tips. p.5
-MS DOS 360K 5¼ inch DSDD Disks — Reading Such Alien Disks p.3
-Famous Computers: The Hewlett-Packard HP2000 and HP3000 Series. . . p.4

TS News: We are experimenting with different newsletter formats and titles. That is one reason for the QL Bulletin launch. A few more launches of variaton on the 8 page newsletter theme are planned, if the time and ambition coincide soon. Of course free newsletters can't be sent out in these times of economic stringency, and there is no risk money to launch these on a subscription basis. Newsletters must be obtained from the user groups sending them out. I can't afford even to answer much correspondence. So, we continue, but of necessity operating on the cheap. The alternative was shutting down the Sinclair newsletters last year, but instead, pieces of the aborted last issue of 1991 were picked out of the discard file and an issue or two was sent out this year. The monthly newsletters had to go and the long mailing list. The reason is not money but economics. As mentioned, donations much over $5 are no longer being accepted to avoid misunderstanding on that count. This is amateur journalism and practice for me at writing/thinking about technical topics. Backissues not kept here -SMUG newsletter is folding but not the group, it is said....Found an ad in Byte dated Apr.1982, p.424 for a hard disk for the ZX-81! ....There is no TS User Group here in Ottawa, local group defunct since 1991....Yes, I do use the ZX-81/TS1000 still, I tested the routines on page 2 on it...... Books: Unofficial ZX-81 LDOS Disk Manual & ZX-81 Tips & Tricks both still being updated, and available for about 35¢ per page still at 160pp & 70pp. approx....Did you see the QL SuperBASIC book published, that was handwritten — That just shows you can publish very simply these days, photocopy or not and certainly don't even need a computer—I use photocopy & typewriter.....

Thought Up In Canada Originally By: Bill Harmer, 97 Ruskin, Ottawa K1Y 4B3

WARNING: The usual disclaimers apply and any of the techniques, circuits, designs, etc. on the hardware article on page 5 certainly can't be guaranteed. Our coverage of Gary Ganger's Ohio T/S column will unfortunately be delayed until the April issue so it can appear with the "Building a Pentode Analogue Computer" article presently being prepared. The article to feature "Wordprocessing on the ZX-81 In Greek" has similarly been delayed.

☞ By the way, I often put a little computer bug at the end of articles to remind everyone that the best thought out article may contain a "bug".

Alphabetical Sort-Indexed Sorts, The Way The Big Boys Do It

In most programs to sort data alphabetically (or numerically, both are covered here) in small RAM computers, the bubble sort or other sort actauuly changes the positions of the elements of data (string array elements like I$(X) for example), in RAM. Bigger computers with big databases like dBASE III (Borland) can't waste the time actually moving these elements physically when they get too large, so they use pointers (an array like N(X)), each pointer pointing to a data array element, and they simply move the pointer list around, until they form an alphabetically in order list when used to point to array elements of data. In dBASE III they do not even physically delete an elment of data, but just change the first byte to "*" to show that name is deleted. The PACK command in dBASE III actually will physically delete the name/record however, to save memory by packing the recrods that way. You can use any sort, quick, shell, or as we show here, a variation on the bubble sort, to do this sort of index sorting, and here is the BASIC listing to do it:

```
10 REM INITIALIZATION-DATA BASE PROG.
20 LET NE=40 (REM no. names/elememts)
30 DIM I$(NE,25) (REM data, 25 chars. each)
40 DIM N(NE)  (REM dimension pointers)
50 REM name bytes 3 to 23, nos. bytes 24,25
```

```
 90 LET OP=1    (sort on name)
100 FOR Z =1 TO NE (REM fill point)
110 LET N(Z)=Z     (ers with array)
120 NEXT Z         (index nos.    )
```

```
149 REM SORT BY INDEX, DATA ARRAY I$(X) SUBROUTINE
150 FAST
160 LET P=0
162 LET E=NE  163 LET G=E  164 LET EN=E 165 REM Loop
170 LET G=INT(G/1.3)
175 IF G<1 THEN LET G=1
180 LET FLG=0
190 FOR X=1 TO E-G
195 LET Y=X + G
200 IF I$(N(X))(1)="*" OR I$(N(X))="<.. 25 spaces..>"
     THEN GOTO 280
201 REM LAST LINE PUT DELETED OR BLANK AT BOTTOM
202 IF OP=1 AND I$(N(X))(3 TO 23)< = I$(N(Y))(3 TO 23)
     THEN GOTO 225
rem sorts first field, name field if OP=1, else--
203 IF OP=0 AND I$(N(X))(24 TO 25) <= I$(N(Y))(24 TO
     25) THEN GOTO 225
204 REM >changed to <= and GOTO 225 NOT GOTO 205 see ① footnote
205 REM EXCHANGE IF OUT OF ORDER ROUTINE
210 LET P=N(X)
215 LET N(X)=N(Y)
220 LET N(Y)=P
224 LET FLG=1
225 NEXT X
250 IF FLG=1 OR G<>1 THEN GOTO 165
260 SLOW
270 RETURN
280 REM PUT UNWANTED, BLANK ELEMENTS AT BOTTOM
281 LET P=N(X)
285 FOR Z=X TO NE-1
290 LET N(Z)=N(Z+1)
295 NEXT Z
296 LET N(NE)=P
297 IF E>1 THEN LET E=E-1
298 LET FLG=1
290 GOTO 250
```

compare differences in ...

```
149 REM ORDINARY SORT SUB.
150 .. ditto as on left
160 LET X$=""
 ... ditto as on left
170    ditto     "
175    ditto     "
180    ditto     "
190    ditto     "
195    ditto as on left
200 IF I$(X)(1)="*" OR
 I$(X)="   ..25 spaces."
 THEN GOTO 280
202 IF OP=1 AND I$(X)(3
 TO 23)<=I$(Y)(3 TO 23)
  THEN GOTO 225
203 IF OP=0 AND I$(X)(24
 TO 25) < =I$(Y)(24 TO 25)
 THEN GOTO 225
205 EXCHANGE IF OUT OF ORD.
210 LET X$=I$(X)
215 LET I$(X)=I$(Y)
220 LET I$(Y)=X$
224 ditto  as on left
225  ditto   "
250 ditto as on left
260  ditto    "
270  ditto    "
280  ditto    "
281 LET X$=I$(X)
285 FOR Z=X TO NE-1
290 LET I$(Z)=I$(Z+1)
295 NEXT Z
296 LET I$(NE)=X$
297  ditto as on left
298  ditto    "
290 ditto     "
```

The index version of the sort is shown on the listing on the left, and the same modified bubble sort is shown in lists non-indexed version on the right for comparison , so you can see what has been changed, to merely rearrange pointers compared to rearranging the items of data, array elements in whole. Slightly different listing methods are needed in the two cases, as shown in the listing of those routines next:

```
600 FOR C=1 TO NE
610 PRINT I$(N(C))(3 TO 23);"=";CODE(I$(N(C))(24));
     "-";CODE(I$(N(C))(25))
620 NEXT C
625 REM last two numbers appear  0-0 to 254-254,
     since one byte and CODE is more efficient to
     store numbers in than reg. char. codes 0-9 only
```

```
600 FOR C=1 TO NE
610 PRINT I$(C)(3 TO 23);
    "=";CODE(I$(C)(24));
    "-";CODE(I$(C)(25))
620 NEXT C
625 REM last two bytes
    are spec. no. code
  (see page 7, data format)....
```

Footnote:

---

① line 204 is GOTO 225 etc. in normal, one condition sort routines in textbooks.

How Do I Read That Darn Disk? Compatibility Problems With Other Computers

How do you read the data off a disk used by/written by another computer than the one you own, or the one to get the data off it? You can try sidestepping the whole thing by using a null modem or some other sort of wire to transfer the data rather than exchanging disks at all. Assuming the easiest case, that of text files that you want to read off another, alien disk only, some of the pointers that follow may help, since the problems are more often software than hardware incompatibility.

The key to reading data off another disk and the point at which a computer will, when not being driven by some sort of special utility with low level (mainly machine code subroutine) access the alien disk initially is in reading off the info on the directory track which includes both the list of text and program files on the disk, usually in ASCII and a table of how the sectors or blocks on the disk are used, often in a non-obvious way. This serves as the table of contents and index of a book does and it is absolutely essential in quick access to the disk contents. It also is the entry point of the normal disk access routines on your computer and the means for the operating system to reject an alien disk as unreadable with its normal methods.

Where is this directory track? On CP/M floppy disks it is normally the first track, on 320K 5¼ inch IBM PC and 720K 3½ inch IBM PC disks it is normally the first track too, although for the 5¼ inch disks this is an older, MS DOS version 1.25 disk standard for formatting, the industry standard for compatibility on the IBM PC and clones being the 360K 5¼ inch disk in which the directory information also runs over to the second track (track 1 versus zero, on the bottom of the disk). Some computers only use the first 35 track/head positions (sometimes called cylinders) like the Radio Shack Color Computer BASIC disks and the Commodore 64 disks, which are also only single-sided. They also put the directory track in the middle of the bands, rather than as the first track, which is on the outside, track 18 for C-64 and 17 for Coco. CP/M favours the standard way, having the directory track on the first track, as in the older MS DOS standard. Unfortunately CP/M likes to put a track map in RAM and operating systems normally don't do that, resulting in lots of incompatibilities when CP/M utilities etc. that use low level disk access calls are run on non-CP/M computers using a software emulator only. One more note: a simple hack when experimenting in disk transfer if you have access to both types of computers involved in the transfer is to put, using low level or track writing calls, the directory track your computer reading the alien disk wants to see on a disk, on the alien disk, ruining some of the data in the process, but perhaps getting quick access to the disk with a minumum of programming. For using an IBM PC clone to read an alien disk, the 320K 5¼ inch format that rather than the 360K will clobber only one track and may be best.

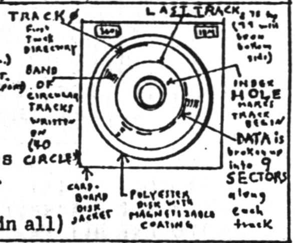
Fig. #1

MS DOS 5¼ inch 360K DSDD Floppy Disk Specs.

VERSION 2.1 —VERSION 3.1 -VERSION 3.2- VERSION 3.3

Track 0, top side
sector: 1-boot m/c &
2.3 -F.A.T. (ORIG.)
4,5 - EXTRA COPY F.A.T.
6.7.8.9 - DIRECTORY (cont)
Track 1, bottom side
sectors 1,2,3 dir.
Track 1, data sec=4-9
Tracks 2-79=data/files

512 bytes per sector
cluster=2 sectors
9 sectors/track (80 in all)


TRACK 0
First Track DIRECTORY
LAST TRACK, # 78 top (79 with been bottom side)
BAND OF CIRCULAR TRACKS WRITTEN ON (40 CIRCLES)
INDEX HOLE MARKS TRACK BEGIN
DATA is broken up into 9 SECTORS along each track
CARD-BOARD DISK JACKET
POLYESTER DISK WITH MAGNETIZABLE COATING

Since the IBM DSDD, 5¼ inch 360K disk is the commonest for non-IBM PC compatible computers to want to read, a summary of its specifications is given in the box above, Fig.#1. Typical contents of the directory track of a disk include as in the IBM PC compatible 360K disk, a boot file, machine code usually with a jump near the beginning to jump over, or simply jump somewhere else, so the next block of info is not executed as machine code and can contain the disk specs, including some info in ASCII easily readable by the hacker using a disk fix/recovery sort of utility. Then there is a list of files on the disk, with some notation usually opposite each one as to the sectors or blocks/tracks where the data of the files is kept on the disk, and a big table of disk sector or block use, called the FAT (File Allocation Table) on the IBM PC, BAT etc. on other computers. Those are the three principal files on a directory track.

Overcoming disk incompatibility is a matter of substituing some form of machine language or low level calls for the normal disk access routines in your computer, since those of your computer normally won't work on an alien disk. File fix/recovery utilities for your computer may have done the hard work already and many computers CP/M and MS DOS especially have full sets of disk conversion utulities out there in commercial, shareware or public domain (usually on BBS systems for the latter).

Disk systems differ in that most use sectors, sections along a track with an identifying header, although some used a whole track at a time. In the IBM and CP/M world 256 or 512 byte sectors were used, a fixed number per track. Of course outer tracks, being circles are larger than inner ones, so some computers like the Commodore 64 use more sectors on the outer than inner tracks although the operating system of that computer has no problem reading a fixed number of sectors (as in C-64, CP/M). IBM PC's take as the smallest snatch of data around a track that can be read in one time — the cluster, which is one or two sectors most often, while most other systems of earlier origin can always read or write to individual sectors due to their DOS program design. Note the existence of hidden files on a disk that do not show up as files on the directory. In a sector system the DOS when it wants to write on a sector, can single out the sector along the track to write on by swooping on the sectors, reading their headings until the wanted one is reached, then switching to read mode (read/write heads are used) and then writing the data to that sector, the operating system allowing some blank bytes on the track for the head to "dribble" random bytes in as it switches from read to write mode electrically. AN AMATEUR PRESS PROD.

Packard is a name that has a certain luxury ring to it, of old American money but Hewlett-Packard is not at all connected to the Packard Automobile and air-engine manufacturer of the 1940's (or to Packard-Bell, a brand of oriental-made IBM compatible computer). It was started in Cupertino California as one of the small, high tech companies that Stanford University Engineering Dept. wanted to encourage by promoting high tech engineering, by Bill Hewlett and David Packard. Perhaps because Hewlett-Packard never achieved the success in computers that tied IBM into the IBM 360 standard of computer architecture for twenty years, its computers are interesting to look at as they tried a number of experiments with the architecture of computers..

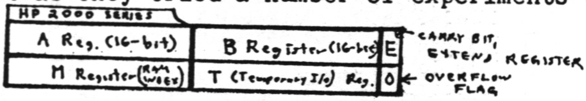


Fig.#1 -Registers of HP2114A 16-bit Minicomputer

Although selling scientific and electronic technician's instruments, the company was into mini-computers in the 1970's, ones not much bigger it would seem in their smaller models to today's microcomputers, or any other bulky, desktop electronic test instrument of that era. For example, the 2114A with 16K bytes of memory used CTL logic and TTL chips, ran at a cycle speed of about ½ MHz and had BASIC, FORTRAN and ALGOL languages available for it. We are not talking microprocessors here, but CPU boards, even though integrated circuits were used. Like many 16-bit minicomputers of the time, this and others in the line, like the physically larger and slightly faster 64Kbyte 2116B, these Hewlett-Packard computers were targeted for the school (meaning college or university mainly), and scientific lab, and industrial process monitoring markets. In short for anyone or rather company or institution that needed a computer but could not afford a big one (like the $100,000-$200,000 IBM 1620, desk-sized computer of the time). The architecture is very similar to the Motorola 6800 microprocessor with an A and B general registers, both 16-bit however, compared to the 6800's 8-bit A and B registers. It had a carry flag which it called an extend register (one-bit) and an overflow flag. It allowed instructions to directly address memory using an index register that it called its M register (M for memory) and all data going to or coming from RAM that way passed through a T register (temporary), which is a bit unique. A simple architecture, quite reminiscent of modern microprocessor lay-out except for using 16-bit registers rather than 8-bit.

A more ambitious one, still havinga large installed base of machines today (1992) is the Hewlett-Packard HP3000 minicomputer. This is a strange computer indeed in that it has no fast, general registers but uses a series of stack pointer registers so that it can use LIFO stacks for everything. All computers use one. Any normal microprocessor or discrete CPU computer uses one for subroutine and interrupt routine handling, when the CPU must stop what it is doing, store the register contants at the time it was interrupted, and use the registers for another job before going back to the job it was originally interrupted in the middle of. This computer, the HP3000, offers advanced and frankly bizarre architecture to try to speed things up, beyond the normal capability of a minicomputer in its class, and of more conventional hardware structure. A further description of the way the HP3000 works will have to be left to another article, since it can't be explained simply very quickly. As an experiment, it has been a success in the sense that many are out there in use, but its architecture has not been that much a success in the sense of all other computer/microprocessor designers having abandoned the normal, general register structure made famous by the IBM 360 in favour of copying Hewlett-Packard's HP3000 type of all-stack structured computer.

The story does not end there. Hewlett-Packard next tried RISC computers, a concept pioneered (and invented even) by IBM but not much exploited by them until rather recently, many years after they invented it. But it was Hewlett-Packard's staying with the concept and pushing that design in a line of their computers that eventually brought everyone else in the industry around to taking a second look at RISC (Reduced Instruction Set Computers). The principle of RISC computers is that having a large set of instructions, means that time is wasted in the CPU or microprocessor handling their complexity and length variations, etc. A reduced size of instruction set meant that the CPU could be designed to run faster. The adavantage of complicated instructions was not worth the time they wasted in overhead if 80% of the instructions were only used 20% of the time. Beyond that, explaining RISC

(Continued next page →)

Hardware Hacker Tricks: Recent Discoveries And Old Gems For PCB Making

In a brief career as a television repair technician, this writer got to open up a lot of tv's and radios, and the things found have become a small mine of information on how to lay out circuit boards. My favourite method is that of the older generation of pcb, using circular, swirling traces, easily put on steel-wool scrubbed single-sided circuit board with a black resist pen (Radio Shack# 276-1530, for example), and also, from experience including some extra 3 point ties and extra-wide traces where possible, both to allow fixes to the circuit in case of errors. An example of such a circuit board is shown in the Fig.#1 illustration. You must also take into account such things as actual componet sizes, perhaps by measuring of the distance between hole for the leads of a resistor or capacitor by placing the component right on the cricuit-board, where you plan to do the drawing in resist. This is certainly a quick if dirty way to make pcboards, but then most hobbyists don't want to waste a lot of time before getting to actually try out their little project.

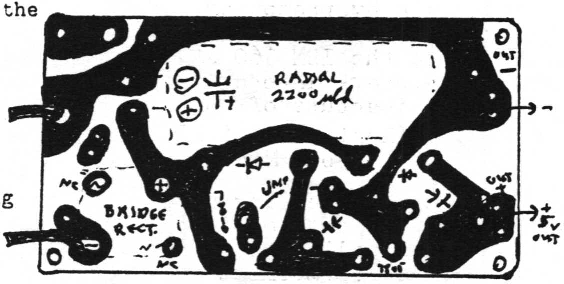
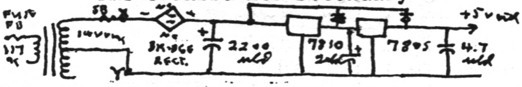


Fig.#1 Example of Circuit Using Resist Pen: + 5 volt from 14 v rms Transformer Secondary



One of the pcboards I designed was a vacuum tube test pcb, and being so long since I had studied tubes and being vague on what the final circuit would be, the design was made flexible to suit later changes in circuit, a virtual universal pcb connected to all pins of the tube socket. That proved a good decision, and wide trace and extra pads with space for three-four lead holes has saved the day many times.

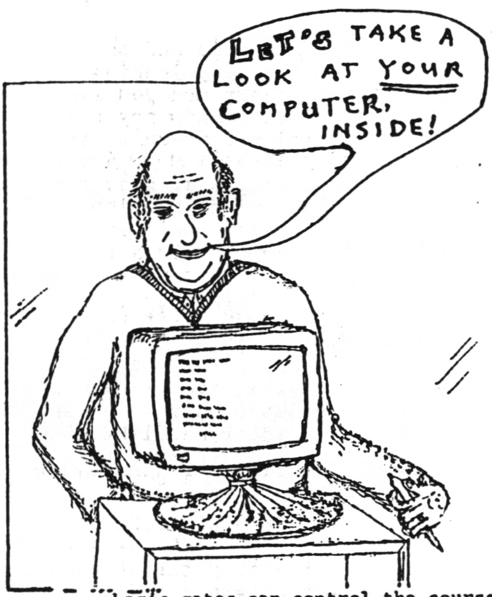
One day at the shop a Hitachi radio came in and on disassembly proved to have little black or brown patches used where resistors would normally be soldered into the circuit board. Attempts to make a resistor compound to paint on pcboards produced some thermistors, but no stable resistors. However if you could buy the compound in sufficiently small quantities, that would be an interesting thing to experiment with. But you can buy nickel and copper paints to use on pcboards, or even to make pcboards from ordinary glass, ceramic, plastic, etc. or make multi-layer pcboards in the style of high tech digital pcbs with electroplated layers.

Another method of miniaturizing circuits is by the use of surface mount components. Small NPN transistors and even integrated cuircuits like the 555, 74LS00, CMOS 4000 and 324 quad-op amp are available in these much smaller packages, to mention just a few of the hobbyist's favourite parts. Radio Shack (USA) sells a package of surface mount resistors (1/8th watt) cheaply too, although you can use normal 1/8th watt resistors if you are skillful. Components are taped to the board with masking tape to hold them during soldering with a small, 15 watt soldering iron. Few hobbyists seem to be using this. You can also photoreduce circuits, as one other user mentions another trick. There is a blank Mylar plastic sheet, TEC-200 available from Meadowlake Co.,P.O.Box 497,Northport,NY,USA 11768, 5 sheets for US$3.95 plus US$1.50 postage, that Rene Bruneau mantions in Sinc-Link, Toronto, as being a medium that you can xerox the image of a pcb design onto, and with an iron set to cotton heat (hot!) you can transfer the image of your pcb design onto the steel wool cleaned copper of the pcboard blank, so that it is ready for etching. Using reducing photocopiers you can reduce (or blow up) artwork, by about 65% in reduction at one step (or the square of the percentage expressed as a fraction, if in two steps, a 70% reduction twice, making about 50% in total). This again allows you lots of thing to try, and if you get the design backwards, you can have it photocopied onto plastic overhead/projection stock, to get the image back the right way (mirror image of the original) by flipping the sheet on the next photocopy. Full and 2 or 3 times size drafting rub-on patterns and resist patterns can be used to make the work smaller, the lines neater, and generally more modern looking than a resist pen, but non-digital circuits should use wide traces to carry the current (rules are given in books). A resist pattern can be copied by photocopy while the black resist is still on, although some touch up with white out will be needed too.

(Continued From p.4 ,HP Computers)

and the use Hewlett-Packard made of them in their RISC computers gets complicated enough to postpone for another time perhaps. But Hewlett-Packard has brought a degree of innovation to computer design that few not actually working with HP computers in their workplace might be aware of. Perhaps it is worth looking into these CPU's further, for learning about them is more than learning about possible radical computer designs only seen in labs by a few. Hewlett-Packard has brought such techniques out into the working world of everyday scientific and business computer use, even if an amateur is not going to be able to afford to own one of these minicomputers at home.

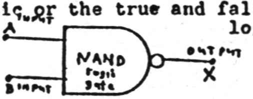
……


LET'S TAKE A LOOK AT YOUR COMPUTER, INSIDE!

A Further Look Inside Computers For The Electronics Student

A digital computer is made up of many small circuit building blocks, arranged exactly to make a computer. The Boolean logic logic gate is one of the most basic. It can process binary numbers (0 or 1 allowed only) to do arithmetic or the true and false of logic.

Fig.#1



This NAND gate can be used to build a whole computer. But it itself is made up of smaller elements, in an electronic computer transistors inside silicon chips. Here next is the exact circuit used for a NAND gate at one time, to do what Fig.#1 shows in logic symbols.

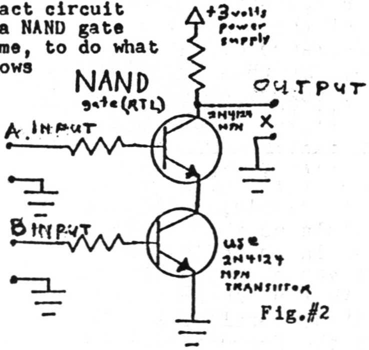


Fig.#2

Logic gates can control the course of calculating or moving data in a computer according to certain conditions fed into the input of a logic gate, a High used as True for example, and a Low (near ground voltage) as saying the condition in question is False. (An example of a condition it may be false "that a number stored in a storage spot like a register is zero." The computer may have to know that, in order to decide if the calculation needs one more step or is finished.)

These logic gates are very useful for they can do arithmetic also. The Fig.#3 circuit shows several NAND gates hooked up together to add two, ~~1bit~~ binary numbers. So Boolean logic and arithmetic can both be run with the same NAND logic gates. So they are simply used for that too inside computers. A third job is to rewire the circuits in the computer to do a different job, something the computer does in the microprocessor or CPU for each new machine language instruction. To save the labour of hand rewiring as done in the old days, the logic gates themselves can rewire or at least make different connections between the wires. In Fig.#4, right, putting a high to input S, will connect data line D to wire B, instead of A. A lot of these circuits can make a computer programmable, the numbers doing the programming called the machine language instructions or simply machine code.

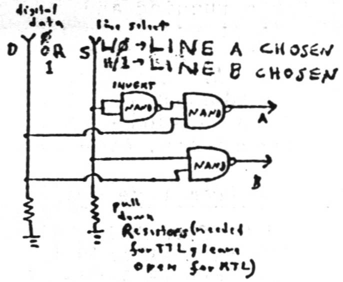


Fig.#4 - line select (deMUX in tech terms)

Fig#3 circuit to add two binary one-bit numbers a_1, a_2

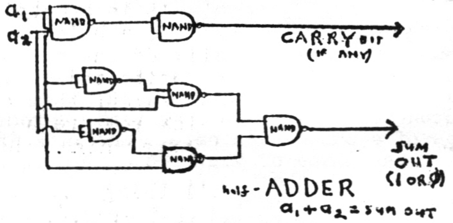


Fig.#3, NAND gates used for adding, binary arithmetic, another important use of the logic gate as a building block for digital computers—from this simplest binary adder circuit, much more complicated ones are built to add more than one binary digit at a time

One more thing that logic gates can be made to do, by building bigger, special purpose circuits of them—to store a number as in RAM memory or fast registers in the microprocessor or CPU. In this NAND gates are connected to form a flip-flop, which will store one binary digit but many flip-flops together can store big numbers, as big as needed.

—The ability to build so many complex electronic circuits with such small parts means the simplest computer will have about 300,000 transistors used to make these circuits today. (Note cost requires NOT and NOR used too.)

B.J.Y. – copyright © June, 1992, in the public domain
– copy freely! –

What Is A Computer? Searching For The Ultimate Answer

If you have ever been asked by a young child what a computer is and then analysed your answer afterwards, you likely have found it to be seriously lacking to your own ways of thinking. What is a computer? Turn to the dictionary and you likely will only find a definition that will satisfy someone who doesn't know any better. It talks most often about high speed, processing of data (and who doesn't from ant to architect, fool to philosopher, so that hardly makes any special claim), arithmetic and logic. So arithmetic only distinguishes it from animals not as clever as "Clever Hans" the circus horse who could tap out answers involving addition. Maybe the heart of such definition is that the computer does only those functions of people or animals. Then call the control circuits hooking a computer to a robot or the engine in your car, something different than being part of the computer itself, as you define computer and you might have the start of a real definition.

Kids are much easier to handle and most adult learners; you just sit them down in front of a computer and show them what they can do with it. Computer is as computer does Today most of this involves using the computer as a word processor, a way of souping up the performance of an electronic typewriter to the point that even a non-typist can use one. So when today's computer user thinks of a computer they think in terms of graphics, boxes, colours, and as well thing like word/character handling, and do not see at all the original definition of a computer as something that computes. We as amateur programmers see it however. Everything that you ask a computer to do is done by computation, rather than estimating; how long a line should be to balance the graphics or line lengths of your essay or personal letter on the screen, and how many lines to display. All calculation. In short it doesn't think the way we do, it just calculates what it is told to, not knowing or particularly caring about the consequences in an individual case. It is figuring blind. The programmer does the caring in advance, trying desperately to figure out the consequences of the particular formula chosen, among other possibilities, to make sure that it will never lead to an unwanted, careless seeming result, no matter what case, even highly unanticipated, that the formula will be used in, say to get all the line lengths to fit on the screen.

So when an article came out recently in the media that talked of a computer not being a calculator in that nowadays it mainly did graphics and calling it a computer as being a word implying calculation (but being a little different, implying more complicated mathematics perhaps) belonging to ancient history of computers, I got mad. It simply is not true. It may not seem to be calculating but inside its intelligence is based on calculation, and often, depending on the application, no more except data handling and a few other details seen only internally.

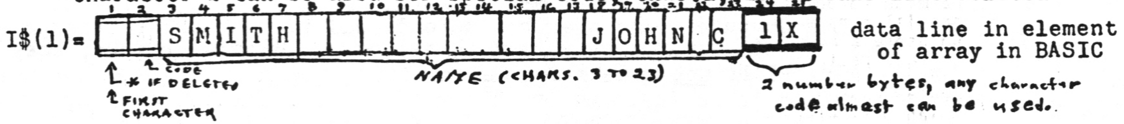
What is a computer? It is many things to many people. To programmers like many of us it is certainly a calculating device. To hardware people it is something else, additionally, but also illustrative to look at. For that reason, a brief perspective of the electronic nature of computers and how they actually do the calculation and program running and data moving, is placed on the page opposite this. Take a look at it if you are curious. But don't spread the malicious rumour that computers don't do any calculation any more. Technical people will think you are a bit silly if you do. And computers don't need to get the same reputation as high school and college graduates of not being able to do mathematics any more even though previous generations did and even seemed to enjoy it. Computers of all things certainly aren't innumerate, even in today's world. TS Hobby Newsletter-B.J.Harmer

……………………

(Continued from page 2-Alpha sort)

The data format in this database used to illustrate the sort is as follows:

- -each data recorde is one line, 25 bytes of a data array (string/Char.) I$(X)
- -there are NE lines of data, the number for NE given in line 20, for DIM's
- -if OP=1, the data is sorted on the name, charcaters 3 to 23
- -if OP=0, the data is sorted on the numbers, two bytes at char.24 & 25, 0-254ea.
- -character 2 can be used for special code, char.1, if * means line deleted



data line in element of array in BASIC

……………………

Both the floppy disk and the hard disk were invented at IBM. It is said that an IBM engineer, eating at a diner, got the idea for computer disks from a juke box.

floppy disk - Is also called a diskette, or in the size our hobby uses (TS), usually, 5¼ inch, a minidisk or minifloppy disk (microfloppy is 3½ inch). (Why floppy? The plastic disk inside the cover is flexible.)

DSDD -This, (double-sided, double-density) refers to a way a disk can be used and formatted, double-sided, meaning data can be written on both the top and bottom of the plastic disk inside the jacket, and double-density meaning double the data per stretch of track as the first floppy disks used (which disks were called single-density)

-For the ZX-81/TS1000 LDOS system you buy 5¼ inch DSDD floppy disks and a 5¼ inch DSDD floppy disk drive, even though the LDOS system itself only writes on the disks in single-density.

-Thus DSDD means the way a disk is used, specifying what to be compatible to.

track -This is a circle traced around the disk as it is rotated inside its cover and disk drive, this circle being a path on the disk that is not seen but is magnetized, in fact by imbedding little permanent magnets lengthwise along the track, in the pattern of your binary data to be stored on that track. Sometimes track means a track-full of data, and often too, at other times a track position (read/write head) opposite a certain track circle/ path. So there are three possible uses for the word in books: the actual circle traced on the disk, a measure of data and the head position.

track position - When all floppy disks were single-sided, then at a given position of the read/write head, out from the centre, only one track full of data could be recorded at that track position. With double-sided disk drives with a read/write head on both the top and bottom of the disk (but connected together so they were always opposite the same track position, or distance from the centre), track path describing the circle along a side of the disk being written on was not the same as the track position. These were often numbered 1 to 40, for such single-sided disks. With double-sided disks you now had two paths to write on at each head position, the top and the bottom track, (track meaning track path on one surface of the disk, either the top or the bottom but not both). For a while the word track was used for either, but that soon led to confusion, so instead of using track position, and often just calling it "track", the word cylinder started being used for track position, and the word track, left to mean only the path on one side of a disk that the read/write head runs along, a circle of course. An LDOS disk has 80 tracks, 0-79. Remember the tracks are circles, not spirals as in an old phonograph.

cylinder - Why call the track position the cylinder? Well, this term comes from the big disk drives like disk packs or hard disks that have more than one disk stacked, so that more than one or two sides can be written on at a given setting (distance from the centre) of the read/write heads, all tied togehter on one arm mechanism. If you think of it, these circular paths stacked one on top of the other, will start to outline a cylinder, if several disks are stacked one on top of another. An LDOS disk has 40 cylinders. (See also above.) 40 cylinders means 40 different positions to move the head assembly to.

block - As mentioned previously, one of the three original meanings of track, was a measure of a chunk of data stored at one place on disk. A track-full of data stored on one track, was then called a track, but is better called a block of data. In LDOS such a block is 1960 bytes including certain header information at the beginning (head of) the track, to identify to the operating system what is stored on that track. See next item for the use of "block" on other disk systems in a different way.

sector - Some systems break a track into sectors that can be written to separately, perhaps 9 or ten of these sections to each track, but LDOS does not use this system of sectors or any sectors. MS DOS, OS/2, CP/M, Radio Shack Color Computer, and the Commodore 64 all use sectors (although the Commodore 64 literature refers to sectors of data as blocks, they mean sectors-full not tracks-full as in LDOS, and not only Commodore).

B.A.T. - Block Allocation Table (also called Track Allocation Table) lets the operating system keep track of which tracks are used to store data and which are empty.